CS 6350.002 Big Data Management and Analytics

Project Report

Project Title: WSDM - KKBox's Churn Prediction Challenge

Team Members:

KeerthiManu Gattu(kxg162530)

Pranathi Peri (pxp162530)

Subrahmanyam Oruganti (oxs160430)

Prasanth Kesava Pillai (pxk163630)

Table of Contents

Introduction

For a business which relies on subscription model, accurately predicting churn rate is a key factor that leads to success eventually. Differences in the prediction has its own side effects.

KKBOX is Asia's leading music streaming service, holding the world's most comprehensive Asia-Pop music library with over 40 million tracks. They use the subscription model to make the media content available to the users. So, predicting the churn rate is very essential for the company. Using survival analysis technique, it tries to predict the residual membership life time for every user. But this technique does not accommodate all the features of a subscription model. So, KKBOX is seeking efficient techniques to predict the churn rate.

Problem Statement

KKBox, a music streaming service provider, has subscribers who can choose to renew or cancel their service when the subscription is about to expire. The users can also opt to auto renewal or choose to cancel the membership at any time. The task here is to predict if a user makes a new service subscription transaction within 30 days after their current membership expiration date. The criteria of "churn" is no new valid service subscription within 30 days after the current membership expires.

Data Description

The dataset we are using for churn prediction is taken from Kaggle competition which can be found at Kaggle. It contains members data and their corresponding transaction logs.

Following are the attributes and(or) features present in the data:

Field	Type	Description
msno	String	This field uniquely identifies a user.
is_churn	Numeric	It is the target variable. A value of '1' indicates churn and a '0' indicates renewal.
Payment_method_id	String	Indicates the type of payment used.
payment_plan_days	Numeric	Indicates the length of membership plan in days.
plan_list_price	Numeric	Indicates the Price of the in New Taiwan Dollar.
actual_amount_paid	Numeric	Indicates the actual amount paid in New Taiwan Dollar.
is_auto_renew	Numeric	Indicates whether a user has opted for autorenewal
transaction_date	Date	Indicates the date on which the transaction has occurred.

membership_expire_date	Date	Indicates the date on which the membership expires.
is_cancel	Numeric	Indicates whether the user cancelled the membership in this transaction.
date	Date	Indicates the date
num_25	Numeric	Indicates number of songs played less than 25% of the song length
num_50	Numeric	Indicates number of songs played between 25% to 50% of the song length
num_75	Numeric	Indicates number of songs played between 50% to 75% of the song length
num_985	Numeric	Indicates number of songs played between 75% to 98.5% of the song length
num_100	Numeric	Indicates number of songs played over 98.5% of the song length
num_unq	Numeric	Indicates number of unique songs played
total_secs	Numeric	Indicates total seconds played
city	Numeric	Indicates city
Bd	Numeric	Indicates age
gender	String	Indicates number
registered_via	Numeric	Indicates the registration method
registration_init_time	Date	Indicates the registration time
expiration_date	Date	Indicates expiration date

Filename	Number of Instances
members.csv	6769473
transactions.csv	21547746
user_logs.csv	392106543
train.csv	992931

Dataset	Number of Instances
trainingRDD	869900
testingRDD	260997

Methodology

1. Pre-processing techniques used

- Mapped values for user_logs, transaction, and members:

  We have four separate files. Each file holds different data of the users. We imported each file into separate RDDs. Then, joining the RDDs with the target variable RDD will create 3 different RDDS. These RDDs help us evaluate which features are to be selected as they are now mapped against the target variable, *is_churn*.

- Decision to choose few features:
  Using the RDDs obtained after joining, we analyzed which features are strongly correlated with the target variable. We first consider only those which are strongly related. Then based on the how the model performs few more features will be included for training.
  For ex: num_25 is a field that has count of the songs that user has played only 25%. Similarly, there are other fields like num_50, num_75 etc. We will not consider all the features in our initial model.

- Removal of outliers:
  The feature bd which indicates the age of the subscriber had outliers. There were 6769473 users in the dataset. Out of them, 4545866 outliers were present. So, we have used decision tree which is built using the features city, bd, registered_via to replace the outliers and used that data in prediction instead of ignoring the rows which have outliers.

- Handling of Categorical Values:
  We had registered_via, is_churn(label) as categorical values. We used StringIndexer to feed these columns as categorical variables to the model.

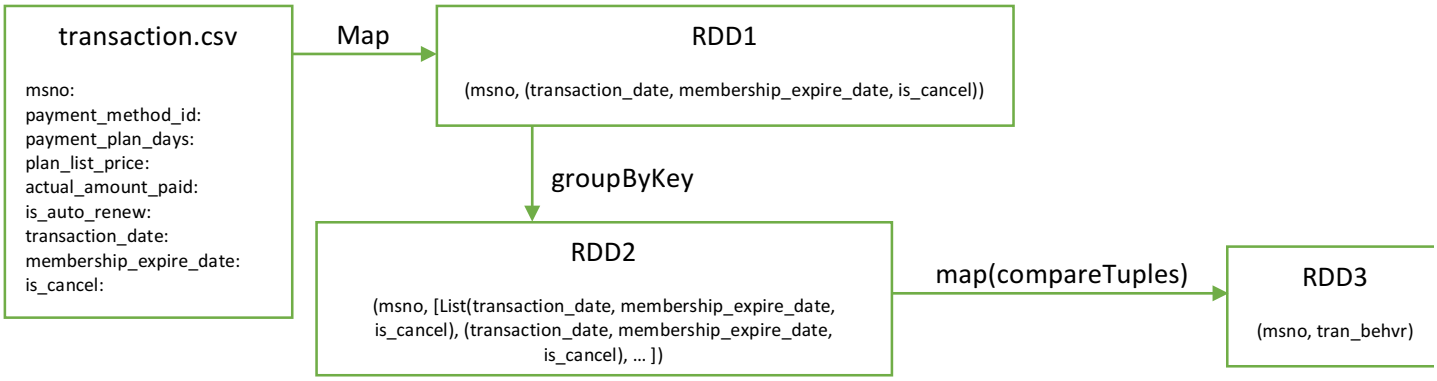
Solutions and Methods

To predict if a user churns or not, we have broken down our approach into three stages. First, we considered the transactional behavior of the subscriber. We mapped the entire transactional behavior of the user into a single field, *tran_behvr*. To do that, we iterated through every transaction of the user to see if the gap between any two transactions is more than 30days(specified). If there exist any two transactions that met the criteria, we have set the value of the field for that particular user to 1 else 0. If the field value is set to 1, then the probability of that user being churned is high.

Then in the second phase we analyzed the user log_data and profile_data. In the third phase, we fed these fields of a user into the classifier and built the model. Technically, the packages in MLlib come into play in the phase 3 to predict *is_churn.*

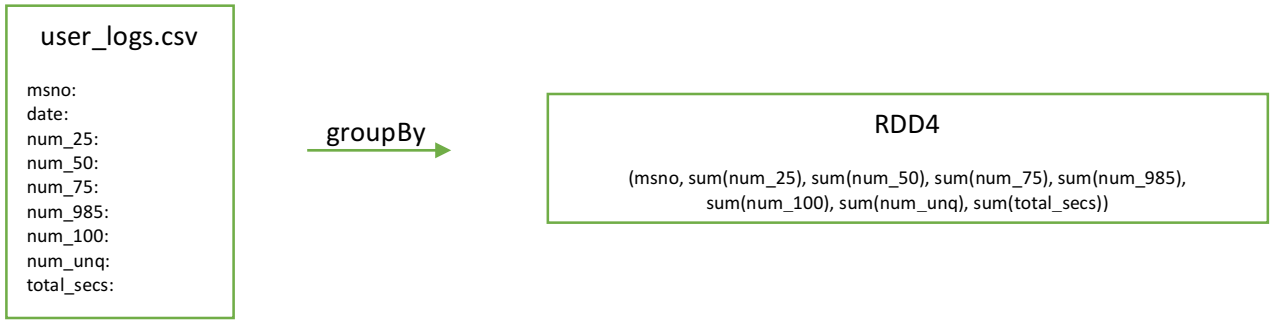
Phase 1:

We leveraged the power of parallel processing in this phase to get all the transaction dates that correspond to a user and to analyze the difference between each transaction. We imported the data from transaction.csv into an RDD and mapped over it to obtain (msno, (transaction_date, membership_expire_date, is_cancel)). Then we grouped by key(msno) to obtain (msno, List[transaction_date, membership_expire_date, is_cancel),…]). Then we have defined a function that checks the difference between current membership_expire_date and next transaction_date of a particular user. If the difference is greater than 30, then tran_behvr is set to 1 else 0.



Phase 2:

In this phase, we have grouped the user_log data. This log data had activity of each user.



Then we imported members.csv into membersRDD and analyzed the columns. As the column *bd* had outliers, we eliminated them using decision tree. For this we considered the rows other than outliers as the training data and built the decision tree. Then we predicted *bd* of the outliers by feeding the other columns to already built decision tree. We had fields *city, registered_via* in members.csv that are to be considered as categorical values when building the model. So, we used StringIndexer to feed these columns as categorical.

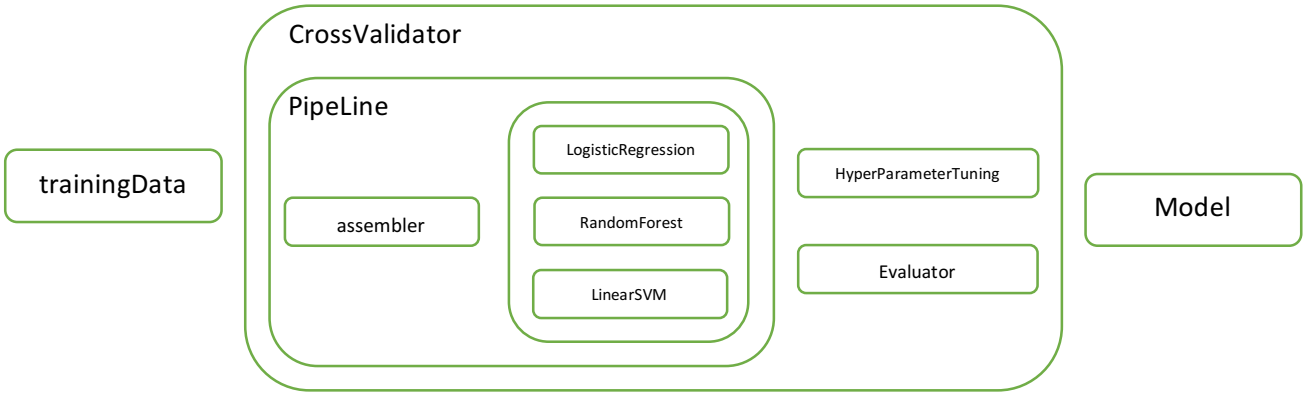
We then imported train.csv into trainRDD(*msno*, *is_churn*). This RDD has *is_churn* which is considered as the label.

Phase 3:

In phase 3, we had the entire data spread into 4 RDDs - RDD3, RDD4, membersRDD, trainRDD. We joined these RDDs to reduce the data into single RDD. So, the final RDD with all the features and label column looked like

RDD(*msno, tran_behvr, sum(num_25), sum(num_50), sum(num_75), sum(num_985), sum(num_100), sum(num_unq), sum(total_secs), registered_via, city, bd,* ***is_churn****)*

Now we have applied various classifiers on the training data as shown below.



Experimental evaluation and Analysis

Classifier	Parameters	ParamGrid	Precision	Recall	Dimensionality Reduction
LogisticRegression	MaxIterations: 10 RegParam: 0.3 ElasticNet: 0.8	lr.threshold - (0.1,0.5,0.6) lr.regParam - (0.1,0.001, 0.001) lr.maxIter - (10,20,30)	0.8861	0.9343	No
LogisticRegression	MaxIterations: 5 RegParam: 0.5 ElasticNet: 0.8	lr.threshold - (0.1,0.5) lr.regParam - (0.1,0.001) lr.maxIter - (10,20)	0.9022	0.9338	No
LogisticRegression	Max Iterations: 10 RegParam: 0.01 ElasticNet: 0.8 Threshold: 0.5	lr.threshold - (0.1,0.5,0.6) lr.regParam - (0.1,0.2,0.01) lr.maxIter - (10,25,30)	0.9032	0.9347	Yes, K = 6
LogisticRegression	Max Iterations: 10 RegParam: 0.5 ElasticNet: 0.8 Threshold: 0.5	lr.threshold - (0.1,0.5,0.6) lr.regParam - (0.1,0.2,0.01) lr.maxIter - (10,25)	0.9020	0.9339	Yes, K = 4
RandomForest	NumTrees: 10 MaxDepth: 2	rf.numTrees - (10,20,30) rf.maxDepth - (3,4,5)	0.9159	0.9346	No
RandomForest	NumTrees: 15 MaxDepth: 4	rf.numTrees - (10,20) rf.maxDepth - (3,4)	0.9097	0.9340	No
RandomForest	NumTrees: 10 MaxDepth: 2	rf.numTrees - (10,20,30) rf.maxDepth - (3,4,5)	0.7238	0.9336	Yes, K = 6
RandomForest	NumTrees: 15 MaxDepth: 3	rf.numTrees - (10,20) rf.maxDepth - (3,4)	0.7385	0.9339	Yes, K = 4
SVM	MaxIterations: 10 RegParam: 0.1	svm.threshold - (0.1,0.5) svm.regParam - (0.1,0.001) svm.maxIter - (10,20)	0.8732	0.9344	No
SVM	MaxIterations: 10 RegParam: 0.5	svm.threshold - (0.1,0.5) svm.regParam - (0.1,0.001) svm.maxIter - (10,20)	0.9020	0.9339	Yes, K = 4

Best results:

1. We obtained best results i.e. Precision – 0.9159, Recall – 0.9346 when we used Random Forest with the following parameters:

NumTrees: 10, MaxDepth: 2, rf.numTrees - (10,20,30), rf.maxDepth - (3,4,5), Without Dimensionality Reduction

Tools and Languages used:

1. Amazon Elastic MapReduce 5.0 (processing)

    We have used a configuration of 1 Master Node and 3 slave nodes each of type m4.xlarge

2. Amazon S3 (storage)

3. Spark 2.2.0 on Hadoop 2.7.3 YARN

4. Scala

5. Zeppelin 0.7.3 (Online notebook that provides environment for Spark)

Related work:

K. Coussement, D. Van den Poel .Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques.Expert Systems with Applications 34 (2008) ,313–327

T.Vafeiadis , K.I.Diamantaras,G.Sarigiannidis, K.Ch.Chatzisavvas. A comparison of machine learning techniques for customer churn prediction

T.Vafeiadis , K.I.Diamantaras,G.Sarigiannidis, K.Ch.Chatzisavvas. Customer churn prediction using improved balanced random forests

Wai-Ho Au , K.C.C. Chan ,Xin Yao.A novel evolutionary data mining algorithm with applications to churn prediction

Anuj Sharma, Dr. Prabin Kumar Panigrahi. A Neural Network based Approach for Predicting Customer Churn in Cellular Network Services

References:

https://spark.apache.org/docs/2.2.0/mllib-linear-methods.html#classification
https://spark.apache.org/docs/2.2.0/mllib-decision-tree.html
https://spark.apache.org/docs/2.2.0/mllib-ensembles.html#random-forests
https://spark.apache.org/docs/latest/sql-programming-guide.html
https://spark.apache.org/docs/2.2.0/ml-classification-regression.html#multinomial-logistic-regression

https://spark.apache.org/docs/2.2.0/ml-pipeline.html

Contributions:

Subrahmanyam Oruganti – Pre-processing, Logistic Regression

Pranathi Peri – Amazon EMR cluster & S3 setup, Random Forest

Keerthimanu Gattu – Amazon EMR cluster & S3 setup, Support Vector Machine

Prasanth Kesava Pillai – Analysis and organization of data

Conclusion:

By analyzing the user log & transaction data, we have predicted whether or not user subscribes back to the service. This type of analysis helps companies analyze what services have to be improved on their end. Also, experimental results showed that the churn prediction model is effective in identifying common customer behavior patterns. The developed system has promising value in the constantly changing subscription industry and can be used as effectively by companies to improve customer relationship and business opportunities.